Data Structures
Homework #3

Due: Nov 29, 2019

1. (5pts) Describe, step by step, the output for the following sequence of deque ADT operations: `addFirst`(3), `addLast`(8), `addLast`(9), `addFirst`(5), `removeFirst`(), `removeLast`(), `first`(), `addLast`(7), `removeFirst`(), `last`(), `removeLast`().

2. (5 pts) Suppose you have a deque D containing the numbers $(1, 2, 3, 4, 5, 6, 7, 8)$, in this order. Suppose further that you have an initially empty stack S. Give a pseudo-code description of a method that uses only D and S (and no other variables or objects) and results in D storing the elements $(1, 2, 3, 5, 4, 6, 7, 8)$, in this order.

3. (15 pts) There is a simple, but inefficient, algorithm, called *bubble-sort*, for sorting a sequence S of n comparable elements. This algorithm scans the sequence $n - 1$ times, where, in each scan, the algorithm compares the current element with the next one and swaps them if they are out of order. Give a pseudo-code description of bubble-sort that is as efficient as possible assuming S is implemented with a doubly linked list. What is the running time of this algorithm (using the Big-Oh notation)?
hint: Avoid index-based operations.

4. (10 pts) When Bob wants to send Alice a message M on the Internet, he breaks M into n **data packets**, numbers the packets consecutively, and injects them into the network. When the packets arrive at Alice's computer, they may be out of order, so Alice must assemble the sequence of n packets in order before she can be sure she has the entire message. Describe an efficient scheme for Alice to do this. What is the running time of this algorithm (using the Big-Oh notation)?
hint: You should be able to achieve $O(n)$ time.

5. (15 pts) Suppose that we need to merge k sorted sequences into one sorted sequence and the total number of elements in these k sequences is n. A straightforward method is to select repeatedly the smallest one from the elements which are the smallest (at the first position) in each of the k sorted sequences. To find the smallest one in these k smallest elements from these k sequences needs $\mathrm{O}(k)$ time. Because there are n elements, the worst case of the approach is $\mathrm{O}(nk)$. Now, we would like to have a method that is better than the straightforward one and can be done in $\mathrm{O}(n \log k)$ time. The idea of the method is depicted in Figure 1, where 8 sorted sequences are given and a data structure is needed to help the merge process. Your job is to provide a binary tree structure that can help merging these k sequences in $\mathrm{O}(n \log k)$ time. Please describe the structure you provide and the process for merging the sorted sequences. In addition, please also explain why $\mathrm{O}(n \log k)$ time can be achieved.

6. (10 pts) Show how to use the ***Euler tour*** traversal to compute the level number of each node in a binary tree T and give a pseudo-code for this approach. What is the running time of the approach.

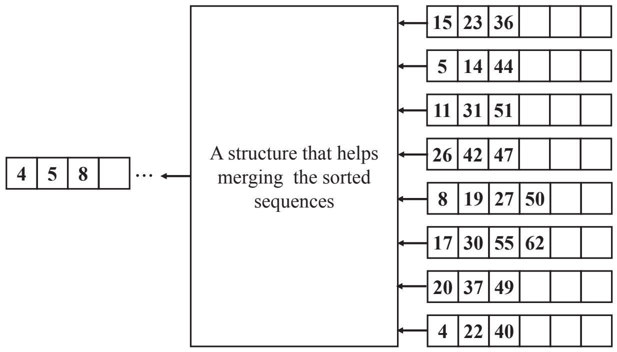

Figure 1: An example of merging 8 given sorted sequences into 1 sorted sequence

7. (40 pts) (**Programming Exercise**)
Univariate polynomial of degree d has the form

$$c_d x^d + c_{d-1} x^{d-1} + \cdots + c_2 x^2 + c_1 x + c_0,$$

where $c_d \neq 0$. The c_i's are the *coefficients*, and $d, d-1, \cdots$ are the *exponents*. By definition, d is a nonnegative integer. In this exercise, we assume that all c_is are integers. We represent each polynomial as a linear list of coefficients and would like to have some operations (functions) on the polynomials. The first node in the list represents the first terms in the polynomial, the second node represents the second terms, and so forth.

Each node contains three fields: the term's coefficient, the term's power, and a pointer to the next term. Write a Python program, that first reads the input file, `inFile.txt`, which has three lines and then performs the indicated operation. The first line is an integer representing the operation defined as below. The second line is the first polynomial and the next line is the second polynomial. The input polynomial, say $4x^3 - 2x + 1$, will be represented as `4x^3-2x+1`. The functions include the following operations:

(1) `add`: Add two input polynomials.
(2) `subtract`: Subtract the second polynomial from the first one.
(3) `multiply`: Multiply two polynomials.
(4) `divide`: Divide the first polynomial by the second one and return the quotient.

The input file thus may be
```
2
4x^3-2x+1
3x^2+x+4
```
Your output will be `4x^3-3x^2-3x-3`.

About submitting this homework

1. For problem 1, 2, 3, 4, 5 and 6, Please

   (1) write all of your solutions on the papers of size A4,
   (2) leave you name and student ID on the first page, and
   (3) hand in your solutions for problem 1, 2, 3, 4, 5 and 6 to me **before** class.

2. For problem 7,
   (1) please finish each problem right after the problem description in the `HW3.ipynb` file provided on the i-school *ischool* (`http://www.ischool.ntut.edu.tw/`) platform; and
   (2) please upload the completed `.ipynb` file with the filename as `HW3_studentID.ipynb` to *ischool* platform.

3. **Late work** is not acceptable. Remember, the **deadline** is the midnight of **Nov 29**, 2019.

4. Honest Policy: We encourage students to discuss their work with the peer. However, each student should write the program or the problem solutions on her/his own. Those who copy others work will get 0 on the homework grade.